數值線性代數 midterm

b05502087 王竑睿

1 Given an m-by-m nonsingular matrix A, how do you efficiently solve the following problems, using Guassian elimination with partial pivoting (GEPP) ?

(a) Solve the linear system $A^k x = b$, where k is a positive number.

algorithms

$$
\begin{aligned}
& A^k x = b \\
\Rightarrow & A(A^{k-1}x) = b \\
\Rightarrow & Ay = b \qquad (\text{令 } y = A^{k-1}x) \\
& \text{由高斯消去法解得y} \\
\Rightarrow & A^{k-1}x = y \\
& \text{重複此步驟逐步減少A的次方}
\end{aligned}
$$

pseudocode

```
function xk = powerA(A,pow,b)
    bk(:) = b(:);
    bk = bk';
    for count=pow:-1:1
        A
        bk
        x = gaussianelim(A,bk);
        A*x-bk
        if(count~=1)
            bk=x;
        end
    end
    xk=x
    %Bk=bk
    %Ak=A
end
```

partial pivoting

```
function x = gaussianelim(A,b);
    [row,col]=size(A);
    n = row;
    x = zeros(n,1);
    for k=1:n-1
        %do partial pivoting
        for i=k+1:n
            if A(i,k) > A(k,k)
                A([i, k], :) = A([k, i], :);
                b([i, k], :) = b([k, i], :);
            end
        end
        for i=k+1:n
            xMultiplier = A(i,k)/A(k,k);
            for j=k+1:n
                A(i,j) = A(i,j)-xMultiplier*A(k,j);
            end
            b(i) = b(i)-xMultiplier*b(k);
        end
    end
    % backsubstitution:
    x(n) = b(n)/A(n,n);
    for i=n-1:-1:1
        summation = b(i);
        for j=i+1:n
            summation = summation-A(i,j)*x(j);
        end
        x(i) = summation/A(i,i);
    end
end
```

required flops

- 高斯消去法使用partial pivoting約有$\frac{2}{3}m^3 + \frac{1}{2}m(m-1)$個flops
- back substitution約有m^2個flops
- 做k次會需要$k \times (\frac{2}{3}m^3 + \frac{1}{2}m(m-1) + m^2)$個flops

(b) Compute $\alpha = c^T A^{-1} b$.

algorithms

$$
\begin{aligned}
& \alpha = c^T A^{-1} b \\
\Rightarrow & \alpha = c^T x \qquad (\text{其中 } Ax = b) \\
& \text{由高斯消去法解得x} \\
\Rightarrow & \alpha = c^T x
\end{aligned}
$$

pseudocode

```
function alpha=cAb(c,A,b)
    x = gaussianelim(A,b);
    alpha = c*x;
end
```

partial pivoting

```
function x = gaussianelim(A,b);
    [row,col]=size(A);
    n = row;
    x = zeros(n,1);
    for k=1:n-1
        %do partial pivoting
        for i=k+1:n
            if A(i,k) > A(k,k)
                A([i, k], :) = A([k, i], :);
                b([i, k], :) = b([k, i], :);
            end
        end
        for i=k+1:n
            xMultiplier = A(i,k)/A(k,k);
            for j=k+1:n
                A(i,j) = A(i,j)-xMultiplier*A(k,j);
            end
            b(i) = b(i)-xMultiplier*b(k);
        end
    end
    % backsubstitution:
    x(n) = b(n)/A(n,n);
    for i=n-1:-1:1
        summation = b(i);
        for j=i+1:n
            summation = summation-A(i,j)*x(j);
        end
        x(i) = summation/A(i,i);
    end
end
```

required flops

- 高斯消去法使用partial pivoting得到x約需要$\frac{2}{3}m^3 + \frac{1}{2}m(m-1)$個flops

- back substitution約有m^2個flops
- 向量c乘以向量x會需要大約$2m$個flops
- 因此總共約需要$\frac{2}{3}m^3 + \frac{1}{2}m(m-1) + m^2 + 2m$個flops

(c) Solve the matrix equation $AX = B$, where B is m-by-n.

algorithms

$$
\begin{aligned}
&\text{將B視為}b_1, b_2, ..., b_n \\
&\text{將X視為}x_1, x_2, ..., x_n \\
\Rightarrow &AX = B\text{可視為n個linear system} \\
&(Ax_i = b_i \text{ for i=1,2,...,n}) \\
\Rightarrow &\text{亦可使用高斯消去法一一解出}
\end{aligned}
$$

pseudocode

partial pivoting

```
function x = gaussianelim(A,B);
    [row,col]=size(A);
    [rowB,colB]=size(B);
    n = row;
    x = zeros(n,colB);
    for k=1:n-1
        %do partial pivoting
        for i=k+1:n
            if A(i,k) > A(k,k)
                A([i, k], :) = A([k, i], :);
                B([i, k], :) = B([k, i], :);
            end
        end
        for i=k+1:n
            xMultiplier = A(i,k)/A(k,k);
            for j=k+1:n
                A(i,j) = A(i,j)-xMultiplier*A(k,j);
            end
            B(i,:) = B(i,:) - xMultiplier*B(k,:);
        end
    end
    % backsubstitution:
    for k=1:colB
        x(n,k) = B(n,k)/A(n,n);
        for i=n-1:-1:1
            summation = B(i,k);
            for j=i+1:n
                summation = summation-A(i,j)*x(j,k);
            end
            x(i,k) = summation/A(i,i);
        end
    end
```

end

required flops

- 高斯消去法使用partial pivoting得到x約需要$\frac{2}{3}m^3 + m(m-1)n + \frac{1}{2}m(m-1)$個flops
- back substitution約有nm^2個flops (B可視為n個行向量)
- 因此總共約需要$\frac{2}{3}m^3 + m(m-1)n + \frac{1}{2}m(m-1) + nm^2$個flops

2 The aim of this problem is to generalize the approach described in class for solving a tridiagonal system of equations to a pentadiagonal system in x $\in R^m$ of the form

$$a_j x_{j-2} + b_j x_{j-1} + c_j x_j + d_j x_{j+1} + e_j x_{j+2} = f_j$$

for $j = 1, 2, ..., m$ with

$$\begin{bmatrix} x_{-1} \\ x_0 \\ x_{m+1} \\ x_{m+2} \end{bmatrix} = \begin{bmatrix} \alpha_{-1} \\ \alpha_0 \\ \alpha_{m+1} \\ \alpha_{m+2} \end{bmatrix}$$

algorithms

此矩陣形如

$$A = \begin{bmatrix} a & b & c & d & e & ... & 0 & 0 & 0 & 0 \\ 0 & a & b & c & d & e & ... & 0 & 0 & 0 \\ 0 & 0 & a & b & c & d & e & ... & 0 & 0 \\ ... \\ 0 & 0 & 0 & ... & 0 & a & b & c & d & e \end{bmatrix}$$

- 此linear system有以下限制

  $x_{-1} = \alpha_{-1}$

  $x_0 = \alpha_0$

  $x_{m+1} = \alpha_{m+1}$

  $x_{m+2} = \alpha_{m+2}$

  但受影響的只有f_1 f_2 f_{m-1} f_m
  可先將f_1 f_2 f_{m-1} f_m
  分別減去$a_1\alpha_{-1} + b_1\alpha_0$、$a_2\alpha_0$ 、$e_{m-1}\alpha_{m+1}$、$d_m\alpha_{m+2} + e_m\alpha_{m+2}$

- 則接下來就可以只考慮

  $x_1, x_2, ..., x_m$

  $A_{(1,1:m)}$,$A_{(2,1:m)}$,...,$A_{(m,1:m)}$

- 利用complete pivoting解$A_{(1:m,1:m)}x_{(1:m)} = f_{(1:m)}$

pseudocode

solve pentadiagonal system

```
function x = penta(A,myalpha,f)
    %[A,myalpha,f]=getMatrix(m);
    m = size(A,1);

    xtemp = zeros(m+4,1);
    xtemp(1) = myalpha(1);
    xtemp(2) = myalpha(2);
    xtemp(m+3) = myalpha(3);
    xtemp(m+4) = myalpha(4);
    f = f - A*xtemp;

    [Arow Acol] = size(A);
    Anew = A(:,3:Acol-2);
    %x = Anew\f;
    x = gaussianelimComplete(Anew,f);
    x = [myalpha(1:2);x;myalpha(3:4)];
end
```

Gaussian elimination with complete pivoting

```
function x = gaussianelimComplete(A,b);
    [row,col]=size(A);
    m = row;
    n = col;
    x = zeros(n,1);
    index = [1:n]';
    for k=1:m-1
        %do complete pivoting
        maxRow = -1;
        maxCol = -1;
        maxElement = -1;
        for i=k:m
            for j=k:n
                if abs(A(i,j)) > maxElement
                    %row change
                    maxRow = i;
                    %column change
                    maxCol = j;
                    %Element
                    maxElement = abs(A(i,j));
                end
            end
        end
        A([maxRow,k],:) = A([k,maxRow],:);
        b([maxRow,k],:) = b([k,maxRow],:);
        A(:,[maxCol,k]) = A(:,[k,maxCol]);
        index([maxCol,k],:) = index([k,maxCol],:);

        for i=k+1:m
            xMultiplier = A(i,k)/A(k,k);
```

```
            A(i,k:n) = A(i,k:n) - xMultiplier*A(k,k:n);
            b(i) = b(i)-xMultiplier*b(k);
        end
    end
    % backsubstitution:
    x(n) = b(n)/A(n,n);
    for i=n-1:-1:1
        summation = b(i);
        for j=i+1:n
            summation = summation-A(i,j)*x(j);
        end
        x(i) = summation/A(i,i);
    end
    %At = A;
    %bt = b;
    toSort = [x,index];
    AfterSort = sortrows(toSort,2);
    x = AfterSort(:,1);
end
```

---

required flops

- 處理f_1 f_2 f_{m-1} f_m約需要12個flop
- 高斯消去法使用complete pivoting得到x約需要$\frac{2}{3}m^3 + \frac{1}{3}m^3$個flops
- back substitution約有m^2個flops
- 因此總共約需要$m^3 + m^2 + 12$個flops